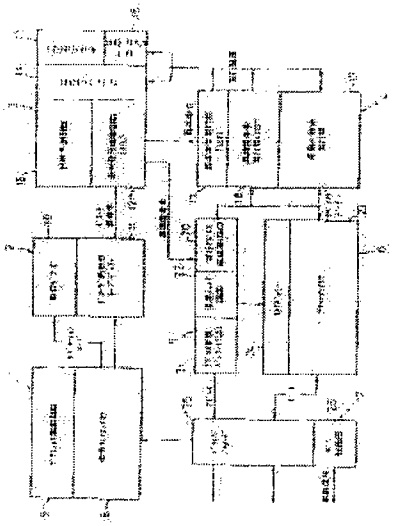
MICROPROCESSOR

Publication number: JP2042534 (A)
Publication date: 1990-02-13
Inventor(s): KISHIGAMI HIDEYA; MIYATA MISAO; OKAMOTO MITSUMASA
Applicant(s): TOKYO SHIBAURA ELECTRIC CO
Classification:
- international: ***G06F9/22; G06F9/38;*** **G06F9/22; G06F9/38;** (IPC1-7): G06F9/22; G06F9/38
- European:
Application number: JP19890083243 19890331
Priority number(s): JP19890083243 19890331; JP19880078207 19880401

Also published as:
JP7120283 (B)
JP2085016 (C)

Abstract of **JP 2042534 (A)**

PURPOSE:To prevent the deterioration of the working efficiency at a prescribed stage of a pipeline by executing the instructions of the 1st and 2nd types in parallel with each other and independently of each other by a pipeline system. CONSTITUTION:The information equivalent to a high function instruction and a basic instruction is read out of an instruction decoding unit 2 and set at an instruction control circuit 12. The high function instruction is generated from the unit 2, and an execution address is calculated by an execution address generating part 20. An address converting buffer 21 performs the conversion of addresses, and the converted address is carried out by a high function instruction executing part 18. The result of execution of the part 18 is written into a future file 14.; At the same time, the basic instruction is generated from the circuit 12 and executed by a basic instruction executing part 17. The result of this execution is written into the file 14. In such a way, the instructions are carried out in parallel with each other. As a result, the disturbance of a pipeline is suppressed and the performance of the pipeline is extremely improved.



Family list
2 application(s) for: **JP2042534 (A)**

1 MICROPROCESSOR

Inventor: KISHIGAMI HIDEYA ; MIYATA MISAO (+1) **Applicant:** TOKYO SHIBAURA ELECTRIC CO

EC: **IPC:** ***G06F9/22; G06F9/38;*** **G06F9/22;** (+3)

Publication info: **JP2042534 (A)** — 1990-02-13
JP7120283 (B) — 1995-12-20
JP2085016 (C) — 1996-08-23

2 Microprocessor

Inventor: KISHIGAMI HIDECHIKA [JP] ; MIYATA MISAO [JP] **Applicant:** TOKYO SHIBAURA ELECTRIC CO [JP]

EC: G06F9/38E; G06F9/38H2; (+1) **IPC:** ***G06F9/38;*** **G06F9/38;** (IPC1-7): G06F9/38

Publication info: **US5155817 (A)** — 1992-10-13

Data supplied from the ***esp@cenet*** database — Worldwide

⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平2−42534

⑤Int.Cl.⁵	識別記号	庁内整理番号	⑬公開　平成2年(1990)2月13日
G 06 F 9/38	370 B	7361−5B	
9/22	350 E	7361−5B	

審査請求　有　請求項の数 7　(全22頁)

⑤④発明の名称　マイクロプロセッサ

⑳特　願　平1−83243

㉒出　願　平1(1989)3月31日

優先権主張　㉜昭63(1988)4月1日㉝日本(JP)㉞特願　昭63−78207

⑦②発明者　岸上　秀哉　神奈川県川崎市幸区堀川町580番1号　株式会社東芝半導体システム技術センター内

⑦②発明者　宮田　操　神奈川県川崎市幸区堀川町580番1号　株式会社東芝半導体システム技術センター内

⑦②発明者　岡本　光正　神奈川県川崎市幸区堀川町580番1号　株式会社東芝半導体システム技術センター内

⑦①出願人　株式会社東芝　神奈川県川崎市幸区堀川町72番地

⑦④代理人　弁理士　三好　秀和　外1名

明　細　書

1．発明の名称

マイクロプロセッサ

2．特許請求の範囲

(1)　デコードされた命令のうち同一の処理過程を経て実行処理される第1の種類の命令をマイクロプログラム制御により実行処理する第1の実行処理手段と、

前記第1の種類の命令と処理過程が異なる第2の種類の命令をハードワイヤード制御により実行処理する第2の実行処理手段と、

デコードされた命令をプログラムシーケンスの順序で発行して、発行した命令を前記第1の実行処理手段および前記第2の実行処理手段のどちらで実行処理するかを選択決定し、前記第1の実行処理手段と前記第2の実行処理手段を独立にしかも並行して動作させる制御手段と

を有することを特徴とするマイクロプロセッサ。

(2)　前記第1の実行処理手段あるいは前記第2の実行処理手段により命令の実行が終了すると、前記第1の実行処理手段あるいは前記第2の実行処理手段よりの実行結果を直ちに書込むための第1の情報保持手段と、

前記制御手段のプログラムシーケンス順序に従って前記第1の実行処理手段および前記第2の実行処理手段よりの実行結果を順序正しく書込むための第2の情報保持手段と

をさらに有することを特徴とする請求項(1)記載のマイクロプロセッサ。

(3)　前記制御手段により発行された命令に関する情報及びプログラムシーケンスにおける命令の実行／終了状態に関する情報を保持し、前記制御手段のプログラムシーケンス順序に従って順次正しく前記第2の情報保持手段を更新するための第3の情報保持手段を

さらに有することを特徴とする請求項(2)記載のマイクロプロセッサ。

(4)　前記第1の種類の命令が、メモリオペランドを有する処理の複雑な高機能命令であり、前記第2の種類の命令が、メモリオペランドを持たな

い基本命令であることを特徴とする請求項(1)に記載のマイクロプロセッサ。

(5) プログラムシーケンスでは先の前記第一の命令実行処理手段で実行処理される第１の種類の命令より、プログラムシーケンスでは後の前記第二の命令実行処理手段で実行処理される第２の種類の命令の方が先に実行処理を終了することを特徴とする請求項(1)に記載のマイクロプロセッサ。

(6) 前記第一の情報保持手段および前記第二の情報保持手段の役割が固定的ではなく、前記第一の情報保持手段の記憶要素および前記第二の情報保持手段の記憶要素対ごとに、ダイナミックにその役割が切替わることを特徴とする請求項(2)に記載のマイクロプロセッサ。

(7) 前記第一の命令実行処理手段で実行処理される命令および、その命令にプログラムシーケンス上で後続する前記第二の命令実行処理手段で実行処理される命令（列）に対して、同一の認識番号を付加する手段と、

前記第一の命令実行処理手段で実行処理中の命令の認識番号を保持する第四の情報保持手段と、

前記第一の情報保持手段の記憶要素および前記第二の情報保持手段の記憶要素の役割等の情報を保持する第五の情報保持手段と、

前記第一の情報保持手段および前記第二の情報保持手段の読み出し／書き込み信号および前記第五の情報保持手段の更新を行う更新手段と

をさらに有することを特徴とする請求項(6)に記載のマイクロプロセッサ。

3．発明の詳細な説明

［発明の目的］

（産業上の利用分野）

この発明は、命令をパイプライン方式により実行処理するマイクロプロセッサに関し、特に、パイプラインの乱れを抑制して、性能を大幅に向上させることができるマイクロプロセッサに関する。

（従来の技術）

近年、マイクロプロセッサにあっては、命令をパイプライン方式より実行処理して、性能の向上を図っている。このパイプライン方式における一般的なステージの構成は、例えば「命令フェッチ→命令デコード→実効アドレス計算→アドレス変換→オペランドリード（読出し）→命令実行→オペランドライト（書込み）」となる（文献「３２ビット・マイクロプロセッサの全容－企業・戦略・技術・市場動向」日経マグロウヒル社，ＰＰ．１３７～１３９）。

このようなパイプライン構成にあって、メモリオペランドを有する高機能命令（Ｉｍ）は、実効アドレスの計算及び実効アドレスから物理アドレスに変換を行うアドレス変換のステージでの処理が必要となる。これに対して、メモリオペランドのない基本命令（ＩＲ）では、上記２つのステージでの処理は不要となる。

したがって、例えば命令のシーケンスが、Ｉｍ→ＩＲ→Ｉｍ→ＩＲ→Ｉｍ→ＩＲのような場合には、パイプラインの"流れ"は、第１２図に示すようになる。なお、各ステージの処理は１サイクルで終了するものとし、命令（Ｉｍ）のオペランドライトをレジスタとして、実行ステージで完了するものとする。また、第１２図において、Ｘ印はステージの動作が休止状態であることを示している。

第１２図から明らかなように、実効アドレス計算のステージ（ＯＡＣ）は、４サイクル目と６サイクル目において休止状態であり、アドレス変換のステージ（ＭＭＵ）は、５サイクル目と７サイクル目において休止状態となっている。

このことから、実効アドレス計算及びアドレス変換の各ステージでの稼働率は、５０（％）となる。

一方、複雑な高機能命令セットを有するＣＩＳＣ（Complex Instruction Set Computer）型のマイクロプロセッサの場合には、実行のステージでの処理に数サイクルを必要とする複雑な高機能命令（Ｉｃ）がある。

このようなマイクロプロセッサにおいて、例えば命令シーケンスが、Ｉｃ→ＩＲ→ＩＲ→ＩＲ→Ｉｍのような場合は、パイプラインの流れが第１

３図に示すようになる。なお、第１３図において、命令Ｉｃは、その実行ステージでの処理に４サイクルかかるものとし、Ｘ印は第１２図と同様とする。

このような場合には、命令Ｉｃの実行に４サイクルかかるために、第１３図から明らかなように、所謂“パイプラインの乱れ”が生じる。これにより、第１３図に示した例では、すべての命令の実行が第１３図の斜線で示した理想的なパイプラインの流れの中で終了せず、３サイクル分（１２サイクル目～１４サイクル目）だけ処理が長くかかっている。

また、高機能命令Ｉｃの実行に４サイクルかかるため、実効アドレス計算（ＯＡＧ）、アドレス変換（ＭＭＵ）及びオペランドリード（ＯＦ）の各ステージにおいて、休止状態が存在することになる。

（発明が解決しようとする課題）

パイプライン処理を行うマイクロプロセッサにおいて、メモリオペランドを有する高機能命令（Ｉｍ）とメモリオペランドのない基本命令（ＩＲ）がそれぞれ交互に実行された場合には、乱れは生じない。しかし、第１２図に示したように、実効アドレス計算及びアドレス変換のステージで、の稼働率が低下するという問題が生じる。

また、実行ステージでの処理に数サイクルを必要とする複雑な高機能命令（Ｉｃ）が実行される場合には、パイプラインの流れに乱れが生じる。これにより、性能が低下するという問題があった。

さらに、このような場合にも、所定のステージでの稼働率が低下することになる。

そこで、この発明は、上記問題に鑑みてなされたものであり、その目的とするところは、ステージの稼働率の低下を防止するとともに、パイプラインの乱れを抑制して、性能を大幅に向上させることのできるマイクロプロセッサを提供することある。

〔発明の構成〕

（課題を解決するための手段）

上記目的を達成するために、この発明に従うマイクロプロセッサは、デコードされた命令のうち同一の処理過程を経て実行処理されるメモリオペランドを有する処理の複雑な高機能命令をマイクロプログラム制御により実行処理する第１の実行処理手段と、前記第１の種類の命令と処理内容が異なるメモリオペランドを持たない基本命令をハードワイヤード制御により実行処理する第２の実行処理手段と、デコードされた命令をプログラムシーケンスの順序で発行して、発行した命令を前記第１の実行処理手段および前記第２の実行処理手段のどちらで実行処理するかを選択決定し、前記第１の実行処理手段と前記第２の実行処理手段を独立にしかも並行して動作させる制御手段とを有している。

そして、この発明に従うマイクロプロセッサは、さらに、前記第１の実行処理手段あるいは前記第２の実行処理手段により命令の実行が終了すると、前記第１の実行処理手段あるいは前記第２の実行処理手段よりの実行結果を直ちに書込むための第１の情報保持手段と、前記制御手段のプログラムシーケンス順序に従って前記第１の実行処理手段および前記第２の実行処理手段よりの実行結果を順序正しく書込むための第２の情報保持手段と、前記制御手段により発行された命令に関する情報及びプログラムシーケンスにおける命令の実行／終了状態に関する情報を保持し、前記制御手段のプログラムシーケンス順序に従って順序正しく前記第２の情報保持手段を更新するための第３の情報保持手段とを有している。

（作用）

上記構成のマイクロプロセッサによれば、高機能命令と基本命令を、それぞれ独立して実行処理するようにして、高機能命令と基本命令が並行してあるいは同時に実行処理されることを可能にしている。

また、制御手段によって発行された命令の実行を、第１の情報保持手段の更新にしたがって開始するようにしている。さらに、メインルーチンからはずれたサブルーチンが実行された後、プログラムシーケンスがメインルーチンに戻った時に、

実行開始の命令を第２の情報保持手段の保持内容にしたがって決定するようにして、命令を再実行できるようにしている。

また、発行された命令に関する情報及び命令の実行／終了状態に関する情報にしたがって、第２の情報保持手段の保持内容がプログラムシーケンス順に順序正しく更新されるようにしている。

（実施例）

以下図面を用いてこの発明の実施例を説明する。

第１図は、この発明を実施したマイクロプロセッサの内部全体の構成を示すブロック図である。

このマイクロプロセッサは、主記憶からの命令データのフェッチを行う命令フェッチユニット（ＩＦＵ）１と、上記命令フェッチユニット１よりの命令データの解読を行うためのデコードユニット（ＤＣＵ）２と、上記デコードユニット２から送られてきた命令情報をその種類すなわち、メモリオペランドを持たない基本命令およびメモリオペランドを持つ基本命令ないしは処理の複雑な高機能命令に従って発行するための命令発行ユニット（ＩＩＵ）３と、命令の実行を上記種類に従ってハードワイヤード制御またはマイクロプログラム制御で行うための命令実行ユニット（ＥＸＵ）４と、メモリオペランドのアドレスを生成するためのメモリ管理ユニット（ＭＭＵ）５と、オペランドデータを管理するためのキャッシュ制御ユニット（ＣＣＵ）６と、上記マイクロプロセッサと外部とのデータ入出力を制御するための入出力部（Ｉ／Ｏ）７とを有している。

上記命令フェッチユニット（ＩＦＵ）１は、主記憶上の命令データ群の一部のコピーを保持する命令キャッシュ・メモリ（Ｉnstruction Ｃache）８や命令キャッシュ・メモリ８への主記憶からの命令データのフェッチ等の制御を行うプリフェッチ制御回路（Ｐrefetcher）９等から構成されるもので、従来と同様のものである。

上記デコードユニット（ＤＣＵ）２は、命令コードの解読を行う命令デコーダ（Ｄecoder）１０やデコードした結果の命令情報を複数個、一時的に保持するデコード済命令ループバッファ（Ｄecoded Ｉnstruction Ｌoop Ｂuffer）１１等から構成される。本実施例ではデコードした命令情報をデコード済命令ループバッファ１１から一度（１サイクル）に２命令分読みだし、命令発行ユニット（ＩＩＵ）３に転送できる構成となっている。

ただし本発明には、デコード済命令ループバッファ１１や一度に２命令分読みだす機能は必ずしも必要ではない。

上記命令発行ユニット（ＩＩＵ）３は、上記デコードユニット２から送られてきた命令情報を、上記種類にしたがって、命令実行ユニット（ＥＸＵ）４ないしメモリ管理ユニット（ＭＭＵ）５に対して発行する命令発行制御回路（Ｉnstruction Ｉssue Ｌogic）１２や汎用レジスタ値を保持するカレントファイル（Ｃurrent Ｆile）１３、フューチャファイル（Ｆuture Ｆile）１４、およびリオーダーバッファ（Ｒeorder Ｂuffer）１５等から構成される。

上記命令発行制御回路（ＩＩＬ）１２は通常のパイプライン処理を行なうマイクロプロセッサがもつパイプライン制御回路の機能（ハザードの検出などを行ない、各パイプライン・ステージの状態制御を行なう）のほかに、上記送られてきた命令情報が、メモリオペランドを持たない基本命令であるかメモリオペランドを持つ基本命令あるいは処理の複雑な高機能命令であるかを選択決定し、後述する複数の命令実行部において上記各命令が並行して実行される様に制御する機能、後述する複数の命令実行部でプログラムシーケンス順とは異って終了する命令実行結果の情報をプログラムシーケンス順に戻すためリオーダーバッファ１５の制御（情報設定／解除）を行なう機能を有する。上記カレントファイル１３はプログラムシーケンス順に従って更新されるが、フューチャファイル１４はプログラムシーケンス順とは無関係に後述する命令実行ユニット（ＥＸＵ）４で実行終了後、その実行結果によってただちに更新される。上記リオーダーバッファ１５は命令実行ユニット（Ｅ

ＸＵ）４の複数の命令実行部でプログラムシーケンス順とは異って終了する命令実行結果の情報を一時保持し、プログラムシーケンス順にカレントファイル１３を更新するためのバッファである。

すなわち、上記基本命令と高機能命令とは実行に要するサイクルが異なり、ここでは、実行に要するサイクルが異なる命令をそれぞれに対応した命令実行部で実行するようにしているため、プログラムシーケンスの順序で発行される命令は、実行が必ずしもプログラムのシーケンスの順序にしたがって終了するとは必らず、順序が逆転することがある。

したがって、リオーダバッファ１５は、プログラムシーケンスの順序でカレントファイル１３の中のレジスタの内容を更新して、上記の逆転した順序をプログラムシーケンスの順序に戻すようにしている。すなわち、Ｏut of order で終了した命令をＲeorderする働きをする。

これにより、割込み等のメインルーチンからはずれたプログラムが実行された場合には、カレントファイル１３の内容を参照することにより、命令を再実行することが可能となる。

また、上記命令発行ユニット（ｉＩＵ）３は、分岐命令の高速実行を行うための分岐予測回路（Ｂranch Ｐrediction Ｌogic）１６等も有する。

上記命令発行制御回路１２、カレントファイル１３、フューチャファイル１４、およびリオーダーバッファ１５は本発明の目的を達成するために必要な構成要件である。ただし、リオーダーバッファ１５を用いないで本発明の目的を達成する方法もあり、それについては他の実施例ということで後述する。また上記カレントファイル１３とフューチャファイル１４は物理的には必ずしも別のものではなくても良く、ひとつのレジスタファイルを２つの部分に分けた場合の一方と他方でも良く、それについても他の実施例ということで後述する。

上記命令実行ユニット（ＥＸＵ）４は、命令の実行をハードワイヤー制御またはマイクロプログラム制御で並行して行なうユニットである。この実施例では、メモリオペランドを持たない基本命令（比較・転送命令・算術・論理演算命令など）をハードワイヤード制御で行なう基本命令実行部（Ｓimple Ｅxecution processor）１７、メモリオペランドを持つ基本命令や処理の複雑な高機能命令をマイクロプログラム制御で実行する命令実行部（Ｉnteger Ｅxecution Ｐrocessor）１８、および浮動小数点演算命令を実行する浮動小数点実行部（Ｆloating Ｅxecution Ｐrocessor）１９の３つの実行部から構成される。

なお本発明は、命令の種類に対応した複数の命令実行部を持つことが特徴であり、必ずしも３つの実行部から構成されなくても良い。また本発明の変形として、メモリオペランドを持たない基本命令の実行部とオペランドの実効アドレス計算を行なう部分を共通化した構成も考えられる。

上記メモリ管理ユニット（ＭＭＵ）５は、メモリオペランドの実効アドレスを生成する実効アドレス生成部（Ｏperand Ａddress Ｇenerator）２０、実効アドレス（論理アドレス）を物理アドレスに変換するアドレス変換バッファ（Ｔranslation Ｌookaside Ｂuffer）２１、メモリ保護のチェックを行う保護チェック回路（Ｐrotection Ｌogic）２２等から構成されるもので、従来と同様のものである。

上記キャッシュ制御ユニット（ＣＣＵ）６は、主記憶上のオペランド群の一部のコピーを保持するデータキャッシュ・メモリ（Ｄata Ｃache）２３や書き込みオペランドデータを一時保持するストア・バッファ（Ｓtore Ｂuffer）２４等から構成されるもので、従来と同様のものである。

上記入出力部（Ｉ／Ｏ）７は、マイクロプロセッサと外部とのデータ入出力を制御する部分でドライバ／レシーバ（Ｄriver ／Ｒeceiver）２５やバス制御部（Ｂus Ｃontrol）２６等から構成されるもので、従来と同様のものである。

第２図は、第１図に示したマイクロプロセッサの内部ブロックにおいて本発明に特に関連する要部ブロックを示したものである。

第２図において、バスは２重線で示し、データ

線は直線で示しており、制御線は省略している。。

そして、第２図における各ブロックの内部をさらに詳細に示すと第３図の如くになる。

第３図において、前記命令発行制御回路（ＩＩＬ）１２は、パイプラインの各ステージで実行中の命令に関する情報を保持するパイプライン・レジスタ（ＯＡＧＲ３０、ＭＭＵＲ３１、ＣＣＵＲ３２、ＩＥＰＲ３３およびＳＥＰＲ３４）と、それらの情報を元にパイプラインの流れを制御するコントロール回路（Ｃｏｎｔｒｏｌ）３５から構成される。パイプラインの流れについては第７図、第８図を参照して後述する。コントロール回路３５はまたリオーダバッファＲＢ１５の制御（データの登録・削除等）も行う。

本実施例では前記命令発行制御回路１２は、１サイクルで２命令分の情報をデコードユニット２のデコード済命令ループバッファ（ＤＩＬＢ）１１から受けることができる。（ただしそのうち１つはメモリ・オペランドを持たない基本命令。）

ＳＥＰＲ３４は現在基本命令実行部１７で実行中の命令に関する情報を保持するレジスタである。

ＯＡＧＲ３０は現在ＯＡＧ２０で実効アドレス計算中の命令に関する情報を保持するレジスタである。

ＭＭＵＲ３１は現在ＭＭＵ５でアドレス変換中の命令に関する情報を保持するレジスタである。

ＣＣＵＲ３２は現在ＣＣＵ６でメモリ・アクセス（オペランドリード）中の命令に関する情報を保持するレジスタである。

ＩＥＰＲ３３は現在ＩＥＰ１８で実行中の命令に関する情報を保持するレジスタである。

なおオペランド・ライトに関する情報は、ＣＣＵ６のストアバッファ２４に保持されるためＩＩＬ１２にはオペランド・ライトに関する情報を保持するレジスタは存在しない。

ＩＩＬ１２の詳細ブロックは第４図参照。

基本命令実行部（ＳＥＰ）１７はメモリ・オペランドを持たない基本命令をハードワイヤー制御で実行する為の演算器（Ａｄｄｅｒ）３６を持つブロックである。演算器３６はＩＩＬ１２中のＳＥＰＲ３４によって直接制御される。

高機能命令実行部（ＩＥＰ）１８は、高機能命令をマイクロプログラム制御で実行するための演算器（ＡＬＵ３７、Ｂａｒｒｅｌ Ｓｈｉｆｔｅｒ38、Ｍｕｌｔｉｐｌｉｅｒ39）および、マイクロプログラムを保持するμＲＯＭ４０およびシーケンサから構成される。ＲＡＬ４１はμＲＯＭ４０のアドレスを保持するためのレジスタ、ＭＩＲ４２はマイクロ命令を保持するためのレジスタ、ＥｒｒＡｄｒ43はエラー発生時のμＲＯＭ４０のアドレスを保持するためのレジスタである。またＳＥＬ４４はＲＡＬ４１、ＥｒｒＡｄｒ43およびＩＩＬ１２のＣＣＵＲ３２のopフィールド８８に保持されている値（次にＩＥＰ１８で実行する命令の先頭マイクロ命令のアドレス）のうちの一つを選択するためのセレクタである。

実行アドレス生成部（ＯＡＧ）２０は、メモリ・オペランドの実効アドレスを算出するための加算器（Ａｄｄｒｅｓｓ Ｇｅｎｅｒａｔｏｒ）４７から構成される。

メモリ管理ユニット（ＭＭＵ）５は、論理アドレス（実効アドレス）を物理アドレスに変換するためのアドレス対を保持するアドレス変換バッファ（Ｔｒａｎｓｌａｔｉｏｎ Ｌｏｏｋａｓｉｄｅ Ｂｕｆｆｅｒ：ＴＬＢ）２１およびメモリ・アクセス権をチェックするためのアクセス権チェック回路（Ｐｒｏｔｅｃｔｉｏｎ Ｌｏｇｉｃ）２２から構成される。

キャッシュ制御ユニット（ＣＣＵ）６は、メイン・メモリのデータの一部のコピーを保持するデータ・キャッシュ（Ｃａｃｈｅ）２３およびライト・データの情報を一時的に保持するストア・バッファ（Ｓｔｏｒｅ Ｂｕｆｆｅｒ）２４より構成される。データ・キャッシュ２３はデータを保持するデータ部（ＤＡＴＡ）４８とアドレスや属性を保持するタグ部（ＴＡＧ）４９から成る。またストア・バッファ２４もデータを保持するデータ部（ＤＡＴＡ）５０とアドレスを保持するアドレス部（ＡＤＤＲＥＳＳ）５１より成る。ＩＥＰ１８より送られてきたライト・データはいったんストア・バッファ２４に格納され、その後データ・キャッシュ２３および主記憶に書き込まれる。

第４図は、第３図中のＩＩＬ１２、ＲＢ１５、ＣＦ１３、ＦＦ１４の部分の詳細図である。

ＤＣＵ２のデコード済命令ループバッファ１１から送られてきた命令の情報は、ＳＥＰＲ３４またはＯＡＧＲ３０に格納される。ＳＥＰＲ３４にはメモリ・オペランドをもたない基本命令の情報のみ格納することができる。一方ＯＡＧＲ３０には全ての命令の情報を格納することができる。

ＳＥＰＲ３４は次のフィールドから構成される。

ＯＰ６０…基本命令の種類を示し（比較、転送、加算など）、ＳＥＰの演算器の機能を制御する。

Ｒ／１６１…ソース・オペランドがレジスタかイミディエイト・データかを区別する。

#src ６２…ソース・オペランドのレジスタ番号を指定する。

#dest６３…デスティネーション・オペランドのレジスタ番号を指定する。

Ｉｍｍ６４…イミディエイト・データ。

ＰＣ６５…命令の先頭アドレス

Ｖ６６…有効ビット

ＯＡＧＲ３０は次のフィールドから構成される。

ＯＰ６７…命令の種類を示す。

Ｒ／ＭＩ６８…ソース・オペランドがレジスタかメモリかを区別する。

#src ６９…ソース・オペランドのレジスタ番号を指定する。

Ｒ／Ｍ７０…デスティネーション・オペランドがレジスタかメモリかを区別する。

#dest７１…デスティネーション・オペランドがレジスタかメモリかを区別する。

Ｉｍｍ７２…イミディエイト・データ。

Ａｍｏｄｅ７３…メモリ・オペランドのアドレッシング・モードを指定する。

Ａｒｅｇ ７４…メモリ・オペランドのアドレッシング・モードで使用するレジスタ番号を指定する。

Ｄｉｓｐ ７５…メモリ・オペランドのアドレッシング・モードで使用するディスプレースメント。

Ｅｘ．７６…その他。

ＰＣ７７…命令の先頭アドレス。

Ｖ７８…有効ビット。

ＯＡＧＲ３０に格納された命令の情報は、命令がパイプラインの各ステージでの処理が進むにつれて、ＯＡＧＲ３０－－＞ＭＭＵＲ３１－－＞ＣＣＵＲ３２－－＞ＩＥＰＲ３３と転送される。

ＯＡＧＲ３０－－＞ＭＭＵＲ３１では、ＯＡＧ２０で、Ａｍｏｄｅ７３、Ａｒｅｇ ７４、Ｄｉｓｐ ７５の情報に基づいて実効アドレス（論理アドレス）の計算が行われる。

ＭＭＵＲ３１－－＞ＣＣＵＲ３２では、ＭＭＵ５で、論理アドレスが物理アドレスに変換される。またメモリ・アクセス権のチェックが行われる。

ＣＣＵＲ３２－－＞ＩＥＰＲ３３では、ＯＰフィールド８８で、μＲＯＭ４０のアクセス（命令を実行する先頭マイクロ命令の読みだし）が行われる。

図中の制御回路（Ｃｏｎｔｒｏｌ）３５は、ＳＥＰＲ３４、ＯＡＧＲ３０、ＭＭＵＲ３１、ＣＣＵＲ３２、ＩＥＰＲ３３に保持されている命令の情報および、以下の信号を入力してパイプラインの状態制御、ハザード検出、リオーダ・バッファ（ＲＢ）１５の制御信号を生成する回路である。制御回路３５の詳細については第５図、第６図を参照して後述する。

ストアバッファ・ビジー信号（Ｓｔｏｒｅ Ｂｕｆｆｅｒ Ｂｕｓｙ）１０２

μプログラム終了信号（μＥＮＤ）１０３

キャッシュ・ミス信号（Ｃａｃｈｅ ｍｉｓｓ）１０４

μ命令でのＧＲへのライト信号（μ－ｖ－ＧＲ）１０５

カレント・ファイル（ＣＦ）１３はプログラムシーケンス順に従って更新される汎用レジスタ値を保持するレジスタ・ファイルであり、フューチャ・ファイル（ＦＦ）１４はＳＥＰ１７／ＩＥＰ１８での命令終了によりただちに更新される汎用レジスタ値を保持するレジスタ・ファイルである。

リオーダ・バッファ（ＲＢ）１５は、ＳＥＰ１７とＩＥＰ１８の２つの命令実行部でプログラムシーケンス順とは異なって終了する命令実行結果を一時保持し、プログラムシーケンス順にＣＦ１３を更新するためのバッファである。本実施例で

はＲＢ１５は８エントリであり、以下のフィールドから構成される。

Ｓtate　１０６…エントリの有効／無効および実行中／実行終了を示す。

Ｒ／Ｍ１０７…命令のデスティネーションがレジスタかメモリかを示す。

＃dest　１０８…デスティネーションがレジスタの場合のレジスタ番号を示す。

Ｒesult　１０９…命令の実行結果を保持する。

Ｆlg　１１０…命令の実行結果のフラグを保持する。

Ｅrror　１１１…命令の実行結果でエラーがあった場合のエラー情報を示す。

ＰＣ１１２…命令の先頭アドレス。

ＲＢ１５への情報の登録は、ＳＥＰＲ３４に保持されている命令がＳＥＰ３４で実行されるタイミングないしは、ＯＡＧＲ３０に保持されている命令がＭＭＵＲ３１に転送されるタイミングで行われる。図中のtail　１１３、head　１１４は各々ＲＢ１５に登録された最も新しい命令情報を保持するエントリ＋１、および最も旧い命令情報を保持するエントリをポイントするレジスタである。ＲＢ１５へは１サイクルでtail　１１３がポイントするエントリおよびtail＋１がポイントするエントリに２命令分の情報を同時に登録できる。またＲＢ１５からは、head　１１４がポイントするエントリのＳtate　１０６が実行終了状態であれば、そのエントリのＲesult　１０９、Ｆlg　１１０に保持されている実行結果にしたがってＣＦ１３およびＦlgレジスタ１１５の値が更新される。またＥrror　１１１にエラー情報がある場合には、μプログラム、シーケンス制御部にエラー信号を発生し、エラー処理のμプログラム・ルーチンを起動する。ＲＢ１５からのデータ読み出しは１サイクルで最大１命令分行うことができる。

ＲＢ１５へ命令情報を登録した時にはtail　１１３は＋１ないし＋２カウントアップされる。またＲＢ１５のデータ読みだしが行われたときはhead　１１４は－１カウントダウンされる。

第５図は、第４図中の制御回路１３５の内部ブロックをしめしたものである。制御回路１３５は、パイプライン・レジスタのレジスタに関する情報を基にハザード・チェックを行う部分と、パイプライン・レジスタの有効信号、ハザード・チェック信号等を基にパイプラインの状態制御を行う状態制御回路（Ｓtate　Ｃontrol　Ｃircuit）１２０から構成される。

図中のハーザード　Ｆ／Ｆ１２１は、１６ビットのレジスタで、パイプライン・レジスタ（ＭＭＵＲ３１、ＣＣＵＲ３２、ＩＥＰＲ３３）の命令が汎用レジスタに結果を書き込むとき、対応するビットに１がセットされていて、この情報を基にハザード検出を行う。ハザード　Ｆ／Ｆ１２１は、ＯＡＧＲ３０のＲ／Ｍ７０、＃dest７１をデコーダ　１２４でデコードした結果でセットされ、ＩＥＰＲ３３のＲ／Ｍ９７、＃dest９８をデコーダ　１２７でデコードした結果でリセットされる。

ＳＥＰＲ３４に保持されている命令がＳＥＰ１７で実行できる条件は、ソース／ディスティネーションに使用するレジスタともに書き変わる可能性がないときである。（すなわちハザード　Ｆ／Ｆ１２１の対応するビットに１がたっていないとき）この条件の検出は、デコーダ　１２２、デコーダ　１２３でＳＥＰＲ３４のＲ／Ｉ６１、＃src６２および＃dest６３をデコードした結果とハザード　Ｆ／Ｆ１２１値とを比較回路ＣＭＰ１　１２８、ＣＭＰ２　１２９で比較しその結果のＯＲ出力信号（ハザード（ＳＥＰ））１３３で行う。この条件が満足されるときハザード（ＳＥＰ）１３３が０となり、満足されないときはハザード（ＳＥＰ）１３３が１となる。

同様にしてＯＡＧＲ３０のＡmode７３、Ａreg７４をデコーダ４　１２５でデコードした結果とハザード　Ｆ／Ｆ１２１の値を比較回路ＣＭＰ３　１３０で比較し、その出力信号（hazard（ＯＡＧ））１３４が０のときＯＡＧ２０で実効アドレスの計算が可能となる。

またＣＣＵＲ３２のＲ／Ｍ８９、＃src ９０をデコーダ５　１２６でデコードした結果とハザード　Ｆ／Ｆ１２１の値とを比較回路ＣＭＰ４　１

３１で比較し、その出力信号（ハザード（ＣＣＵ））１３５が０のときソース・オペランド（レジスタ）の読み出しが可能となる。

状態制御回路（State Control Circuit）１２０は、上でのべた３つのハザード信号（hazard（ＳＥＰ）１３３、hazard（ＯＡＧ）１３４、hazard（ＣＣＵ）１３５）、パイプライン・レジスタの有効信号（Ｖ（ＩＥＰ）１０１、Ｖ（ＣＣＵ）９６、Ｖ（ＭＭＵ）８７、Ｖ（ＯＡＧ）７８）やレジスタ／メモリ信号（Ｒ／Ｍ（ＩＥＰ）９７、Ｒ／Ｍ１（ＭＭＵ）８９、Ｒ／Ｍ２（ＭＭＵ）９１）およびＩＩＬ１２外部からの信号（ストア・バッファビジー信号１０２、μＥＮＤ１０３、Cache miss １０４、μ－ｖ－ＣＲ１０５）を入力して、パイプラインの状態制御を行う以下の信号を出力する。

ＳＥＰ－１３６…ＳＥＰＲに保持されている命令がＳＥＰで実行可能なとき１になる。

ＯＡＧ－ＭＭＵ１３７…ＯＡＧＲに保持されている命令が次サイクルでＭＭＵＲに進めるとき１になる。

ＭＭＵ－ＣＣＵ１３８…ＭＭＵＲに保持されている命令が次サイクルでＣＣＵＲに進めるとき１になる。

ＣＣＵ－ＩＥＰ１３９…ＣＣＵＲに保持されている命令が次サイクルでＩＥＰＲに進めるとき１になる。

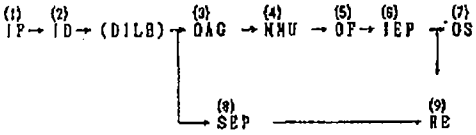
ＩＥＰ－ＳＢ１４０…ＩＥＰＲに保持されている命令が次サイクルでストアバッファに情報を転送するとき１になる。

第６図は、第５図中のステートコントロール回路の具体的な回路例である。

次に、第１図を参照して上記、本発明に従うマイクロプロセッサのパイプライン処理動作の概略について説明する。

すなわち、パイプライン処理動作の概略は以下の様になる。

（以下余白）



(1)　ＩＦ（命令フェッチ）ステージ

ＩＦＵにおいて、命令キャッシュメモリ８からの命令のフェッチを行うステージ。

(2)　ＩＤ（命令デコード）ステージ

ＤＣＵにおいて、命令デコーダ１０で命令のデコードを行い、内部命令フォーマットに変換する。なお内部命令フォーマットに変換された命令はデコード済命令ループバッファ１１に格納される。そして、内部命令フォーマットはメモリ・オペランドを持たない基本命令とメモリ・オペランドを持つ基本命令ないしは高機能命令の２種類あり、命令発行制御回路（ＩＩＬ）１２によってそれぞれ発行される。

(3)　ＯＡＧ（オペランド実効アドレス算出）ステージ

ＯＡＧ２０のアドレス発生回路４７で、命令発行制御回路（ＩＩＬ）１２によって発行されたメモリオペランドを持つ命令のメモリ・オペランド実効アドレス（論理アドレス）を算出するステージ。

(4)　ＭＭＵ（アドレス変換）ステージ

ＭＭＵ５のアドレス変換バッファ２１で、メモリ・オペランドの論理アドレスを物理アドレスに変換するステージ。また保護チェック回路２２でメモリ保護のチェックも行われる。

(5)　ＯＦ（オペランド・フェッチ）ステージ

ＣＣＵ６のデータ・キャッシュメモリ２３からメモリ・オペランドを読み出すステージ。また、レジスタオペランドの読み出しも行われる。

(6)　ＩＥＰ（命令実行）ステージ

ＥＸＵ４の高機能命令実行部（ＩＥＰ）１８において、μプログラム制御でメモリオペランドを持つ基本命令あるいは高機能命令を実行するステージ。

(7)　ＯＳ（オペランド・ストア）ステージ

ＩＥＰ１８での実行結果をＣＣＵ６のストア・

バッファ２４に書き込むステージ。ただしこのステージがあるのは、命令のディスティネーションがメモリの場合のみ。なお演算結果はストア・バッファ２４を介して、データ・キャッシュメモリ６とマイクロプロセッサ外部の主記憶に、パイプライン処理とは非同期に書き込みが行われる。

(8)　ＳＥＰ（命令実行）ステージ

ＥＸＵ４の基本命令実行部（ＳＥＰ）１７において、ハードワイヤード制御で命令発行制御回路（ＩＩＬ）１２によって発行された基本命令を実行するステージ。なおＳＥＰ１７で実行される命令は、メモリ・オペランドを持たない基本命令のみ。

(9)　ＲＥ（リオーダ）ステージ

ＩＥＰ１８およびＳＥＰ１７よりの実行結果をリオーダバッファ（ＲＢ）１５によりリオーダしてカレントファイル（ＣＦ）１３に書込むステージ。

以上のパイプライン処理のうち、(8) ＩＥＰを除く他のステージは、基本的には１サイクルでその処理が終了する。ただしキャッシュ・ミス、ＴＬＢミスが生じたときには、(1) ＩＦ、(4) ＭＭＵ、(6) ＯＦのステージの処理も複数サイクル必要となる。また、ハザード（例えば、ＩＥＰステージの実行結果を実効アドレス算出に使用する等）が生じたときには、いわゆる"待ち"が生じて１サイクルで処理が終了しなくなる。

本発明の特徴は、複数の命令実行部を持ち命令の並列実行を可能とすることである。すなわち本実施例では、主に、ＳＥＰステージおよびＲＥステージが新らたに加わった点が従来技術と比べて新しい。

次に、第７図および第８図を参照して、上記本発明の特徴的な処理動作をさらに詳細に説明する。

第７図および第８図は、本発明の実施例（すなわち、基本命令実行部ＳＥＰ１７がある場合）のパイプラインタイミング例をそれぞれ示し、第７図のタイミング例は、第１２図に示した従来のタイミング例に対応し、第８図の例は、第１３図の従来のタイミング例に対応する。

ただし第７図、第８図では簡単のために、デコード済命令ループバッファ（ＤＩＬＢ）１１以降の部分のみ示し、命令はＤＩＬＢ１１中に有るものと仮定する。

第７図は、命令シーケンスが

Im1 ⟶ IR2 ⟶ Im3 ⟶ IR4 ⟶ Im5 ⟶ IR6

の場合のパイプライン・タイミング例－１である（Ｉｍ）と基本命令（ＩＲ）は（ＤＩＬＢ）から１サイクルで同時に読み出されたものとする。またＩｍ１とＩｍ３はディスティネーションがレジスタ、Ｉｍ５はディスティネーションがメモリとし、ハザードは生じないものとする。　サイクル１ではＩｍ１，ＩＲ２の２命令分の情報がＤＩＬＢ１１から読み出され、ＩＩＬ１２のＯＡＧＲ３０およびＳＥＰＲ３４レジスタにセットされる。

命令Ｉｍ１は、サイクル２で命令発行制御部１２によって発行され、実行アドレス生成部（ＯＡＧ）２０によって実行アドレス算出が行われ、サイクル３でアドレス変換バッファ（ＴＬＢ）２１によってアドレス変換が行われ、サイクル４でメモリ管理ユニット（ＭＭＵ）５によってオペランドフェッチが行われ、サイクル５で高機能命令実行部（ＩＥＰ）１８によって実行され、サイクル６でディスティネーションがレジスタのためフューチャーファイル（ＦＦ）１４へその実行結果が書き込まれる（第７図のＦＦの欄の↑Ｉｍ１を参照）。

一方、これと並行して、基本命令ＩＲ２は、サイクル２で命令発行制御部１２によって発行され、基本命令実行部（ＳＥＰ）１７によって実行され、サイクル３でフューチャーファイル１４へその実行結果が書き込まれる（第７図ＦＦの欄の↑ＩＲ２を参照）。

ここで、リオーダバッファ（ＲＢ）１５への命令情報の登録は、高機能命令Ｉｍは、実行アドレス算出ステージで行われ、基本命令ＩＲは、基本命令実行部１７での実行ステージで行われるため、Ｉｍ１およびＩＲ２の情報は、図示する如くサイクル３で登録される。第７図のＲＢの欄の命令の

上の“×”および“●”印は、命令が各々“実行中”および“実行終了”であることを示している。

一方、リオーダバッファ１５からカレントファイル（ＣＦ）１３への命令実行結果の書き込みは、リオーダバッファ（ＲＢ）１５において命令情報が削除されたサイクルで行われる。従って、Ｉｍ１の場合は、その命令情報がサイクル７でリオーダバッファ１５から削除されているため、サイクル７で、その実行結果が、カレントファイル（ＣＦ）１３へ書き込まれる。また、ＩＲ２の場合は、その命令情報が、サイクル８でリオーダバッファ１５から削除されているため、サイクル８でその実行結果が、カレントファイル（ＣＦ）１３へ書き込まれることとなる。

すなわち、フューチャーファイル（ＦＦ）１４は、命令実行後ただちに更新（書込み）されるため、プログラムシーケンス順とはなっていないが、カレントファイル（ＣＦ）１３は、リオーダバッファ１５から命令情報が削除されるタイミングで更新されるためプログラムシーケンス順に命令実行結果がファイルされている。

命令Ｉｍ３、ＩＲ４、Ｉｍ５、ＩＲ６の場合も、上述したと同様に処理されるものである。

第８図は、命令シーケンスが

Ic1 ーー＞IR2 ーー＞IR3 ーー＞IR4 ーー＞Im5

の場合のパイプライン・タイミング例－２である。この例の場合もＩｃ１の実行に時間がかかっているが、その間にＩＲ２、ＩＲ３、ＩＲ４の実行は基本命令実行部（ＳＥＰ）１７で先に終了している。

すなわち、高機能命令Ｉｃ１は、サイクル２～４で、実行アドレス算出、アドレス変換、およびオペランドフェッチが行われ、サイクル５～８で高機能命令実行部（ＩＥＰ）１８によって実行され、サイクル９でフューチャファイル（ＦＦ）１４へその結果が書き込まれる。

一方、これと並行して、基本命令ＩＲ２は、サイクル２で基本命令実行部（ＳＥＰ）17によって実行され、サイクル３でフューチャファイル（ＦＦ）１４へその実行結果が書き込まれる。

ここで、リオーダバッファ（ＲＢ）１５からカレントファイル（ＣＦ）１３への命令実行結果の書き込みは、リオーダバッファ（ＲＢ）１５において命令情報が削除されたサイクルで行われる。

従って、第７図に示した例と同様に、カレントファイル（ＣＦ）１３には、プログラムシーケンス順に命令実行結果がファイルされるものである。

以上、第７図および第８図の例からわかるように、本発明では複数の命令実行部を持ち命令の並列実行することにより、従来例で生じていたパイプラインの乱れを押さえ、また各パイプライン・ステージの稼働率の低下を押さえることができ、結果として大幅な性能向上を得ることができる。

また、通常の命令実行状態において、フューチャーファイル（ＦＦ）１４に保持されている汎用レジスタ値は、カレントファイル（CF）１３に保持される汎用レジスタ値と異なっている。これはプログラム・シーケンス順では後のメモリ・オペランドを持たない基本命令が、プログラム・シーケンス順では前の高機能命令実行部（ＩＥＰ）１８で実行されるメモリ・オペランドを持つ高機能命令より先に基本命令実行部（ＳＥＰ）１７で実行され、フューチャーファイル（ＦＦ）１４を更新するためである。ただし、高機能命令実行部（ＩＥＰ）１８で実行した命令でエラー（割込み）が発生した場合には、命令の再実行を保証するためにフューチャーファイル（ＦＦ）１４の値をカレントファイル（CF）１３の値に戻さなければならない。このためにカウンタ１１９が用意されている。割込み処理μプログラム・ルーチンではカウンタ１１９を利用してカレントファイル（ＣＦ）１３の値をフューチャーファイル（ＦＦ）１４にコピーすることができる。

第９図に発明の実施例を適用したＭＰＵと周辺ＬＳＩから成るシステム構成例を示す。この例はＶＭＥバス２００につながる比較的簡単なシステムであり、以下のＬＳＩ、ＩＣから構成される。

ＭＰＵ２０１

ＩＣＴ２０２…割込みコントローラ

ＣＧ２０３…クロック・ジェネレータ

メモリ…ＳＲＡＭ（０ウェイト　３２Ｋバイト）
２０４
ＥＰＲＯＭ（０ウェイト　３２Ｋバイト）
２０５
ＤＲＡＭ（３ウェイト　４Ｍバイト）
２０６
通信インタフェース…セントロニクス　１チャネル２０７
ＲＳ２３２Ｃ　２チャネル
２０８
その他…Ｔ／Ｒ　トランシーバ／レシーバ２０９
Ｂｕｆ　バッファ２１０，２１１
Ｄｅｃｏｄｅ　アドレス・デコーダ２１２

本発明を使用したＭＰＵを使用したシステム構成は、従来のＭＰＵを使用したシステム構成と何ら変わるところはない。すなわち本発明を使用したＭＰＵを使用することによりシステム・レベルで必要な付加回路は無く、高性能なシステムを構築することができる。

次に、第１０図および第１１図を参照して本発明に従うマイクロプロセッサの第２実施例について説明する。

前述した本発明の第一の実施例では、リオーダバッファ（ＲＢ）１５を用いることにより、本発明の目的を達成したが、第２実施例ではリオーダバッファ（ＲＢ）１５を用いないで本発明の目的を達成するようにしている。

第１０図は、第一の実施例の第４図に対応するものであり、第一の実施例と同じ要素には同じ番号をつけてある。第一の実施例と第二の実施例の違いは、次の通りである。

まず、第二実施例は、第一の実施例の構成要素であるリーダバッファ（ＲＢ）１５を削除した構成となっている。そして、第一の実施例のカレントファイル（ＣＦ）１３とフューチャファイル（ＦＦ）１４は、第二の実施例では一つの汎用レジスタファイル３０２になっている。ただしそのエントリ数は１６の＜Ｘ＞パートおよび１６の＜Ｙ＞パートの合計３２エントリからなる。第二の実施例では、フラグ（FLG0-FLG3）、エラー情報（Erroro-3）、プログラムカウンタ（PC0-3）を一時的に保持する４エントリのステータスファイル３０１および汎用レジスタファイル（ＧＲ）３０２への書込み／読み出し信号を生成するＧＲコントロール回路（ＧＲ　Control）３０３を新たに加えている。

以下、第二の実施例の特徴について説明する。

第２の実施例の汎用レジスタ３０２は先に述べたように、１６の＜Ｘ＞パートおよび１６の＜Ｙ＞パートの合計３２エントリからなる。＜Ｘ＞パートおよび＜Ｙ＞パートは、第１の実施例のカレントファイル（ＣＦ）１３とフューチャファイル（ＦＦ）１４のように、一つの汎用レジスタＲｉに対して２本のレジスタ（Ｘｉ，Ｙｉ）を用意している。ただし第一の実施例と異なる点は、＜Ｘ＞パートがカレントファイル、＜Ｙ＞パートがフューチャファイルと固定的でなく、＜Ｘ＞パートのＸｉがカレントの値を保持しているレジスタなら、対応する＜Ｙ＞パートのＹｉがフューチャの値を保持しているレジスタ、あるいはＹｉがカレントの値を保持しているレジスタなら、対応するＸｉがフューチャの値を保持しているレジスタ、と言う様に、各汎用レジスタＲｉに対し２本のレジスタ（Ｘｉ、Ｙｉ）がダイナミックにその役割が切り替わることである。

例えば、ある瞬間の＜Ｘ＞のパート、＜Ｙ＞パートのＸｉ、Ｙｉの役割は次のようになっている。

カレントレジスタ値：　X0 X1 X2 Y3 X4 X5 Y6 Y7 Y8 X9 X10 X11 X12 X13 Y14 Y15

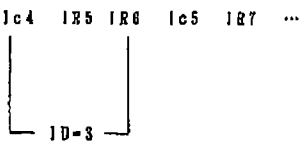
フューチャレジスタ値：Y0 Y1 Y2 X3 Y4 Y5 X6 X7 X8 Y9 Y10 Y11 Y12 Y13 X14 X15

第１１図は、第１０図中のＧＲコントロール回路（ＣＲ　Control）３０３の内部ブロックを示したものである。第１１図中には４つのＩＤレジスタ（３０４～３０７）、＋１回路３０８およびＧＲアドレス生成回路（ＧＲ　address generator）３０９から構成される。ＧＲアドレス生成回路３０９は、パイプラインレジスタの汎用レジスタのアクセスに関する情報（３１０～３１３）およびＩＤレジスタの値（３１４～３１６）を入力

して汎用レジスタファイル３０２の読み出し／書込みアドレス信号（３１８～３２１）を出力するブロックである。ＧＲアドレス生成回路３０９には汎用レジスタファイル３０２の状態を示す３つのフリップフロップ群（３２２～３２４）がある。

いま、第一の実施例で示した様なパイプライン構成の場合には、プログラムの命令シーケンスと命令実行順序が逆転するのは、"Ｉｃ命令に続くＩＲ命令列"であるただしＩｃはメモリオペランドを持つ命令ないしは実行ステージに数サイクル要する複雑な命令（高機能命令）を示し、ＩＲはメモリオペランドを持たない基本命令を示す。またこの様なパイプライン構成の場合には、ＩＲ命令は最大４つのＩｃ命令を飛び越して先に終了する可能性がある。例えばいま命令シーケンスが

（先頭）



の場合で、Ｉｃ１の命令実行ステージのサイクル数が大きい場合、ＩＲ１，ＩＲ２はＩｃ１より先に実行が終了し、ＩＲ３はＩｃ１，Ｉｃ２より先に実行が終了し、ＩＲ４はＩｃ１～Ｉｃ３より先に実行が終了し、ＩＲ５、ＩＲ６はＩｃ１～Ｉｃ４より先に実行が終了することになる（ただしハザードが生じない場合）。またＩＲ７はＩｃ１の実行が終了するまで実行されない。

この場合問題となるのは、例えばＩｃ１命令実行中に例外が発生した場合、ＩＲ１～ＩＲ６の実行により更新される汎用レジスタおよびフラグ、ＰＣなどのステータスを元に戻す必要があることである。このためにＩＲ１～ＩＲ６の実行結果は、まず汎用レジスタファイル３０２のカレントの値を保持しているレジスタ（例えばＸｉ）の対のレジスタ（例えばＹｉ）に書込み、またフラグ、ＰＣなどのステータスもステータスファイル３０１に一時書き込む。そしてＩＲ命令直前のＩｃ命令が実行ステージを終了するサイクルでＩＲ命令の結果を保持しているＸｉとＹｉの役割を切り替える。例えばこの例の場合、Ｉｃ１が実行ステージを終了するサイクルでＩＲ１とＩＲ２の結果を保持しているＸｉとＹｉの役割を切り替える。

この方法の利点はＩｃ命令に後続するＩＲ命令はハザードが生じないかぎり、いくつでも先行して実行することができ、第一の実施例に見られた様なリオーダバッファ１５のエントリ数による制限が生じないことである。またカレントの値を保持しているレジスタは、ＧＲアドレス生成回路３０９中のＦ／Ｆ群によってＸｉまたはＹｉの切り替えを行うため、Ｉｃ命令の実行終了時に複数命令のＩＲの実行が終了している場合にその結果を１サイクルで更新する（すなわちＸｉとＹｉの役割を１サイクルで切り替える）ことができる。

次に具体的にどのようにしてＸｉとＹｉの役割を切り替え、汎用レジスタファイル３０２の読み出し／書込みを制御するかについて説明する。

Ｉｃ命令とそれに続くＩＲ命令に対して０～３のＩＤ番号を割当てる（前述の命令シーケンス列参照）。第１１図中のＩＤレジスタ（３０４～３０７）は各々パイプラインレジスタ（３０～３３）中に保持されている命令のＩＤ番号を保持している。また汎用レジスタに対して以下の３つのフリップフロップ（Ｆ／Ｆ）×１６のＦ／Ｆ群を設ける。

すなわち、フューチャＦ／Ｆ群３２２と、有効Ｆ／Ｆ群３２３と、ＩＤ　Ｆ／Ｆ群３２４とである。

フューチャＦ／Ｆ群（Ｆuture Ｆ／Ｆ群）３２２は、１６個のＦ／Ｆで、汎用レジスタファイル３０２のＸｉがカレントの値を保持しているとき対応するフューチャＦ／Ｆｉは１、Ｙｉがカレントの値を保持しているときフューチャＦ／Ｆｉは０となる。

有効Ｆ／Ｆ群（Ｖalid　Ｆ／Ｆ）３２３は、１６個のＦ／Ｆで、フューチャの値（フューチャＦ

／Ｆｉの値が１の時Ｙｉ、０の時Ｘｉの値）が有効な時１、そうでないとき０となる。

ＩＤ　Ｆ／Ｆ群３２４は、１６個で２ビットのＦ／Ｆで、フューチャの値が有効なとき、その値を書き込んだ命令のＩＤ番号を示す。

ＧＲアドレス生成回路３０９は、これらＦ／Ｆ群（３２２～３２４）の値、パイプラインレジスタの汎用レジスタのアクセス情報（３１０～３１３）およびＩＤレジスタの値（３１４～３１６）をもとに汎用レジスタファイル３０２の読み出し／書込み信号（３１８～３２１）やＦ／Ｆ群の値の更新の制御を次のようにして行う。

１．ＳＥＰ１７で実行されるＩＲ命令の実行に必要なソースオペランドのレジスタＲｉ（ＳＥＰＲ３４の#src６２で指定される）は、対応する有効Ｆ／Ｆｉ＝１の時は、フューチャの値（フューチャＦ／Ｆｉの値が１の時Ｙｉ、０の時Ｘｉの値）、有効Ｆ／Ｆｉ＝０の時は、カレントの値（フューチャＦ／Ｆｉの値が１の時Ｘｉ、０の時Ｙｉの値）とする。

２．ＳＥＰ１７で実行されるＩＲ命令の実行結果を格納するディスティネーションのレジスタＲｉ（ＳＥＰＲ３４の#dest６３で指定される）は、先行命令が無い（実行が終了している：Ｖ７８＝Ｖ８７－Ｖ９６－Ｖ１０１＝０）場合には、カレント（フューチャＦ／Ｆｉの値が１の時Ｘｉ、０の時Ｙｉ）、そうでないときにはフューチャ（フューチャＦ／Ｆｉの値が１の時Ｙｉ、０の時Ｘｉ）とする。

３．ＯＡＧ（実行アドレス算出）ステージに必要な汎用レジスタＲｉ（ＯＡＧＲ３０のＡmode７３、Ａreg ７４で指定される）は、対応する有効Ｆ／Ｆｉ＝１の時は、フューチャの値（フューチャＦ／Ｆｉの値が１の時Ｙｉ、０の時Ｘｉの値）、有効Ｆ／Ｆｉ＝０の時は、カレントの値（フューチャＦ／Ｆｉの値が１の時Ｘｉ、０の時Ｙｉの値）とする。

４．ＩＥＰ（命令実行）ステージに必要なソースオペランド（ＣＣＵＲ３のＲ／Ｍ１８９、#SRC９０で指定される）は、対応する有効Ｆ／Ｆｉ＝１の時は、フューチャの値（フューチャＦ／Ｆｉの値が１の時Ｙｉ、０の時Ｘｉの値）、有効Ｆ／Ｆｉ＝０の時は、カレントの値（フューチャＦ／Ｆｉの値が１の時Ｘｉ、０の時Ｙｉの値）とする。ただし有効Ｆ／Ｆｉ＝１の時でも対応するＩＤＦ／Ｆ≒ＩＤ４　３０７の時は、ソースオペランドの読み出しは待たされる。

５．ＩＥＰ（命令実行）ステージでＩｃ命令が終了する時には、そのＩｃ命令と同じＩＤ番号を持ち、なおかつ有効Ｆ／Ｆｉ＝１の汎用レジスタＲｉのフューチャＦ／Ｆｉの値を反転し、また有効Ｆ／Ｆｉを０にリセットする。

６．ＳＥＰ１７で実行されるＩＲ命令の実行結果のレジスタＲｉ（ＳＥＰＲ３４の#dest６３で指定される）のフューチャ（フューチャＦ／Ｆｉの値が１の時Ｙｉ、０の時Ｘｉ）が、このＩＲ命令と異なるＩＤ番号≒ＩＤ　Ｆ／Ｆｉの場合）には、このＩｒ命令の実行は待たされる。

７．ＳＥＰ１７で実行されるＩＲ命令の実行結果のフラグ（Ｆｌｇ）、エラー情報（Ｅｒｒｏｒ）およびＰＣは、ステータスファイル３０１のＩＤ１レジスタ３０４の値３１７で示されるエントリーに一時書き込まれる。そのエントリ番号と同じＩＤ番号のＩｃ命令の実行終了時にそれらの値がＦｌｇ１１５，Ｅｒｒｏｒ１１６、ＰＣ１１７にセットされ、更新される。

以上のようにして汎用レジスタファイル３０２の読み出し／書込み信号（３１８～３２１）やＦ／Ｆ群の更新の制御を行うことにより、比較的簡単なハードウェアで、本発明の目的を達成することができる。

従って、第一実施例の場合、割込みが発生した場合に、フューチャファイル１４の値をカレントファイル１３の値に戻す必要があり、これに最低１６サイクルを必要で（汎用レジスタが１６本の場合）、これがオーバーヘッドとなり性能低下の原因となっていたが、第二実施例の場合、１つの汎用レジスタ３０２で行っているため、割り込みの発生に対しても、値の移し換えの必要がないものであり、性能低下は起こらない。

また、第一実施例の場合、プログラムシーケンス順で後続する命令が先行する命令を飛び越して実行できる命令数は、リオーダバッファ15のエントリ数によって制限される。すなわちエントリ数が小さければ性能が低下し、またエントリ数を大きくするとハード量が増加してしまう。

それに対し、第二実施例の場合は、一つの汎用レジスタ302においてXパートとYパートの役割を切り替えて書込み読み出しを制御しているため、飛び越して実行できる命令数を大きくすることができる。

また、第一実施例の場合、高速分岐の手法として分岐予測を行う場合には、分岐予測が失敗した場合に汎用レジスタの値を元に戻すのに最低16サイクルを必要で、これがオーバーヘッドとなり性能低下の原因となっていたが、第二実施例の場合、汎用レジスタの値を元に戻す必要がないものである。

[発明の効果]

以上説明したように、この発明によれば、第1の種類の命令と第2の種類の命令を、パイプライン方式によりそれぞれ独立して並列実行処理するようにしたので、パイプラインにおける所定のステージでの稼働率の低下を防止するとともに、パイプラインの乱れを抑制することが可能となる。これにより、性能を大幅に向上させたマイクロプロセッサを提供することができる。

4. 図面の簡単な説明

第1図は、本発明を実施したマイクロプロセッサの内部全体構造を示すブロック図、

第2図は、第1図に示したマイクロプロセッサにおける要部ブロック図、

第3図は、第2図に示すブロック図の各ブロックの内部をさらに詳細に示したブロック図、

第4図は、第3図におけるIIL、RB、CF、FFの詳細図、

第5図は、第4図における制御回路の詳細図、

第6図は、第5図に示す状態制御回路の詳細図、

第7図および第8図は、本発明の実施例におけるパイプライン処理動作のタイミング図、

第9図は、本発明の実施例を適用したMPUと周辺LSIから成るシステム構成図、

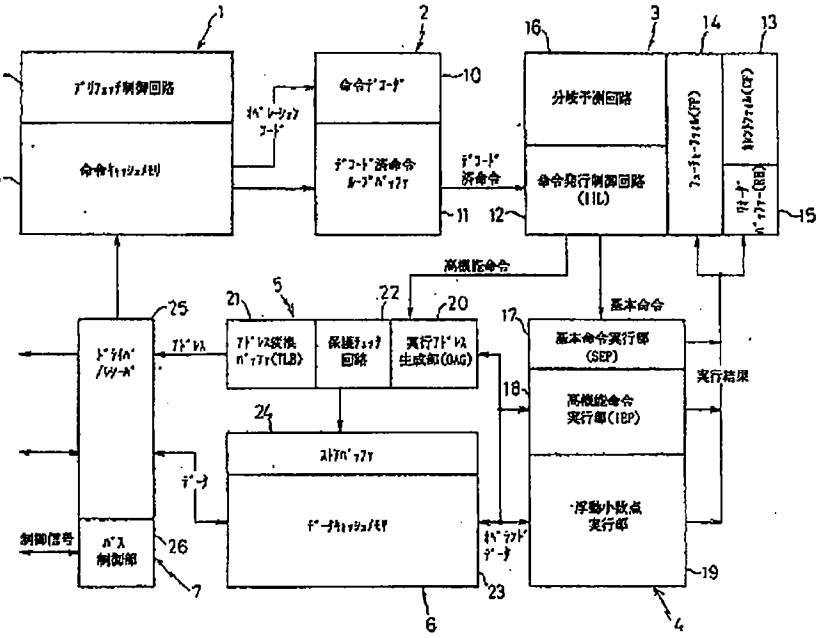
第10図は、本発明に従うマイクロプロセッサの第二実施例の要部構成図、

第11図は、第10図におけるGRコントロール回路の詳細図、

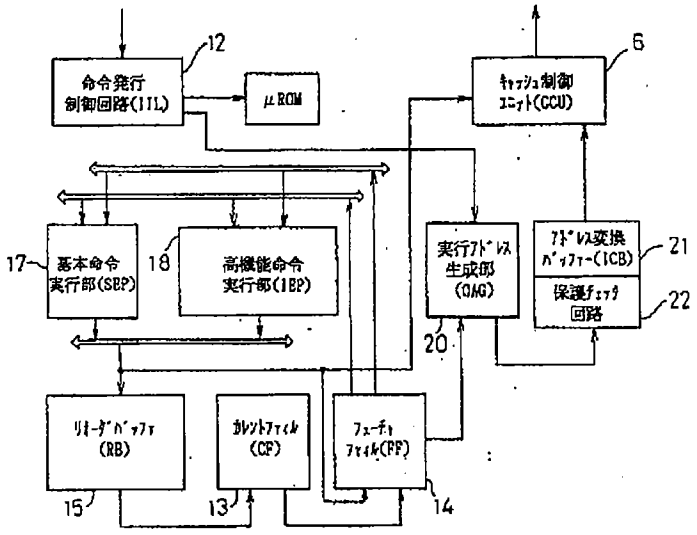
第12図および第13図は、従来例におけるパイプライン処理動作のタイミング図である。

1…命令フェッチユニット（IFU）
2…デコードユニット（DCU）
3…命令発行ユニット（IIU）
4…命令実行ユニット（EXU）
5…メモリ管理ユニット（MMU）
6…キャッシュ制御ユニット（CCU）
7…入出力部（I／O）
10…命令デコーダ
11…デコード済命令ループバッファ（DIL）
12…命令発行制御回路（IIL）
13…カレントファイル（CF）
14…フューチャファイル（FF）
15…リオーダバッファ（RB）
17…基本命令実行部（SEP）
18…高機能命令実行部（IEP）
20…実行アドレス生成部（OAG）
21…アドレス変換バッファ（TLB）
23…データキャッシュメモリ

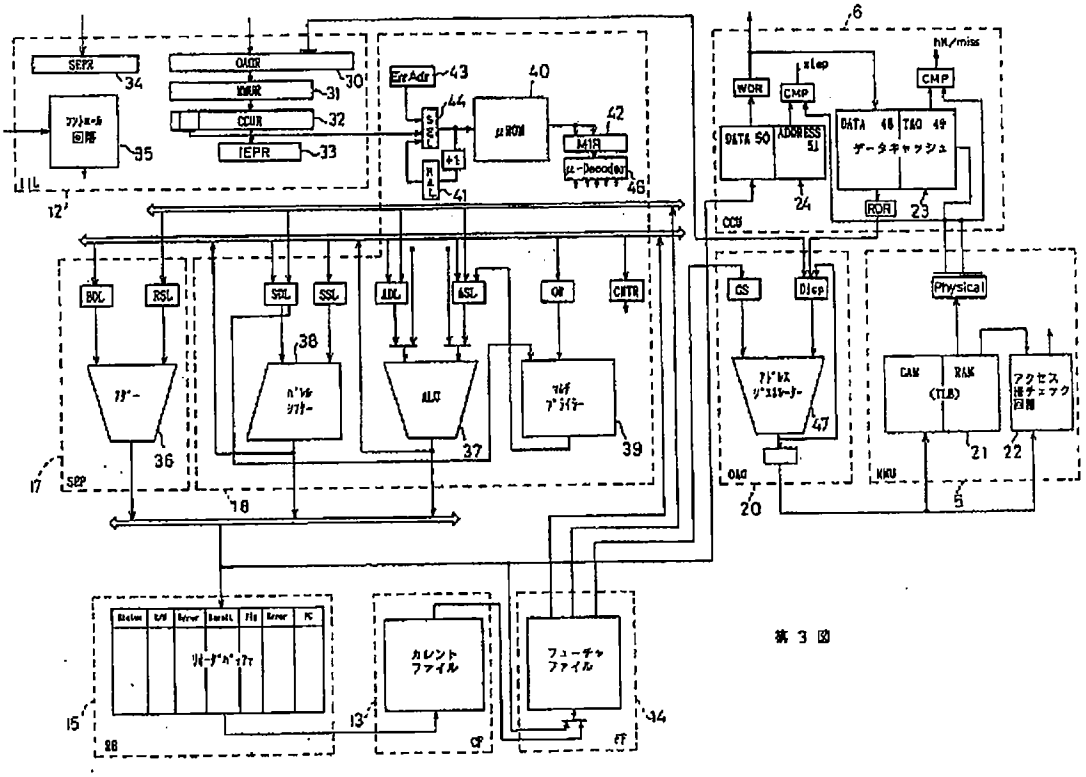
代理人弁理士 三好秀和



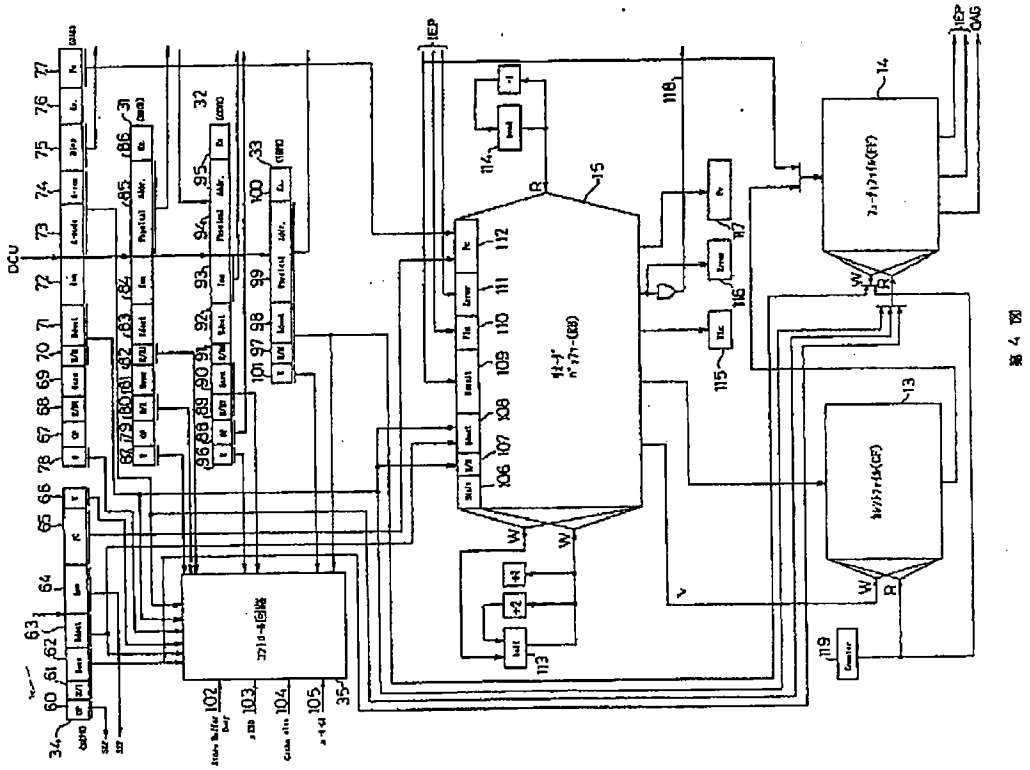
第 1 図



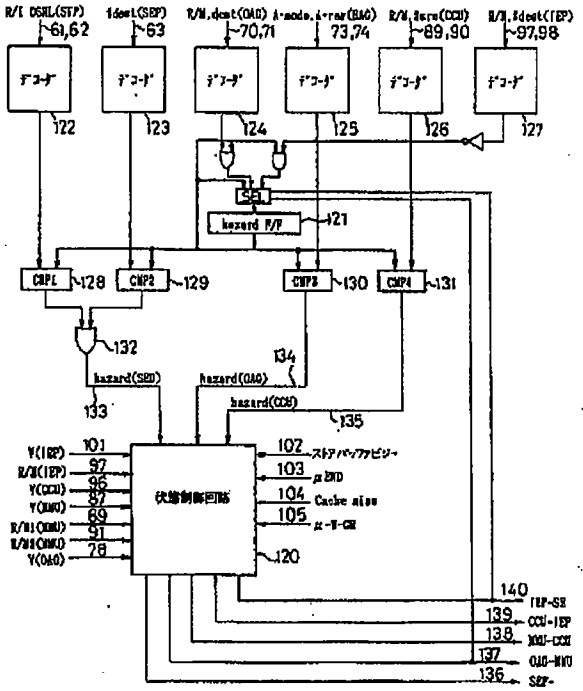
第 2 図



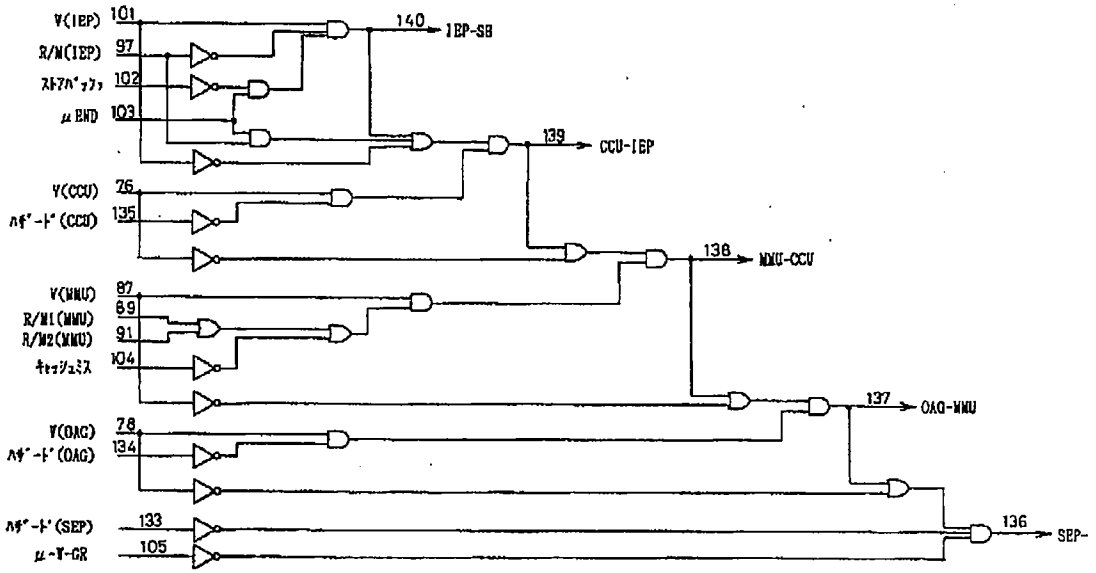
第 3 図



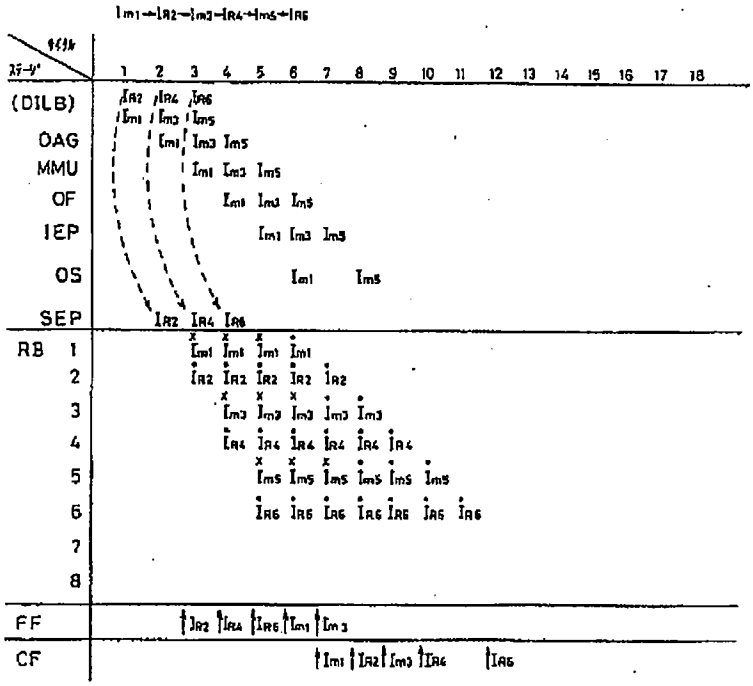
第 4 図


61,62
63
70,71
73,74
89,90
97,98
122
123
124
125
126
127
121
128
129
130
131
132
133
134
135
101
97
96
87
69
91
78
102
103
104
105
120
140
139
138
137
136

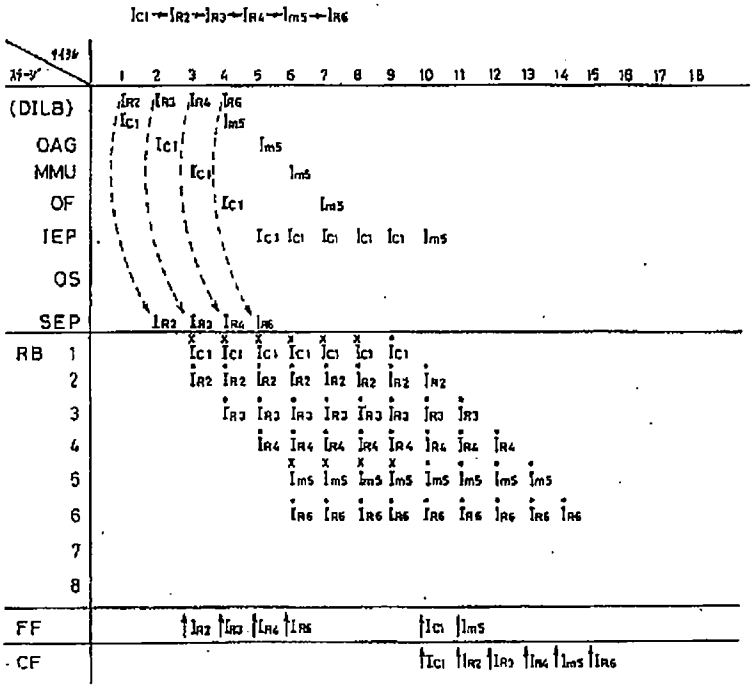
第 5 図


V(IEP)
101
R/W(IEP)
97
102
μ END
103
140
139
V(CCU)
76
135
138
V(MMU)
87
69
91
104
137
V(OAG)
78
134
133
105
136

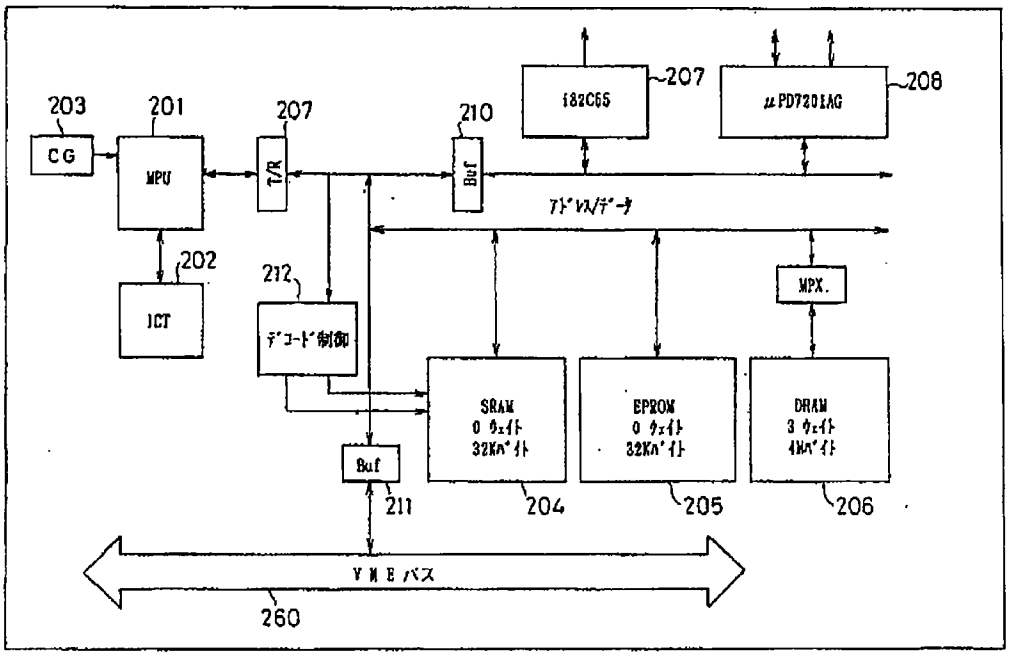
第 6 図


Im1→IR2→Im3→IR4→Im5→IR6
サイクル
ステージ
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
(DILB)
OAG
MMU
OF
IEP
OS
SEP
RB 1
2
3
4
5
6
7
8
FF
CF

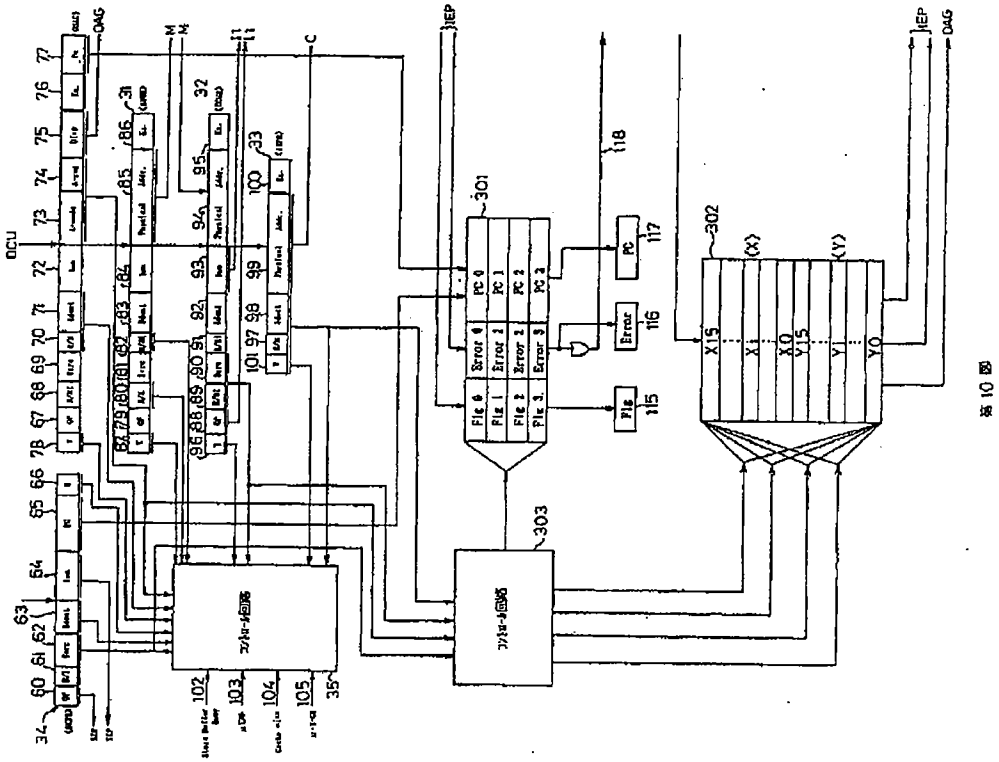
第7図


IC1→IR2→IR3→IR4→Im5→IR6
サイクル
ステージ
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
(DILB)
OAG
MMU
OF
IEP
OS
SEP
RB 1
2
3
4
5
6
7
8
FF
CF

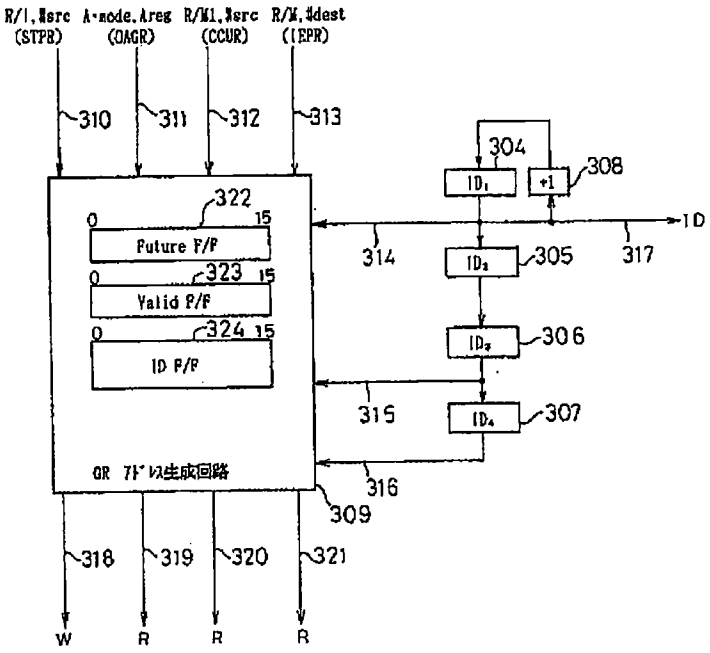
第8図


203
201
207
210
CG
MPU
T/R
Buf
82C65
207
μPD72001AG
208
202
ICT
212
MPX.
SRAM
0 ｳｪｲﾄ
32Kﾊﾞｲﾄ
EPROM
0 ｳｪｲﾄ
32Kﾊﾞｲﾄ
DRAM
3 ｳｪｲﾄ
4Mﾊﾞｲﾄ
Buf
211
204
205
206
260

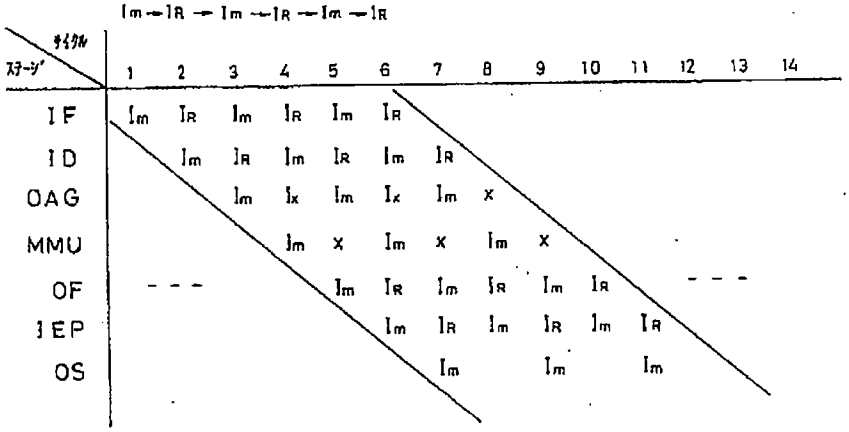
第 9 図



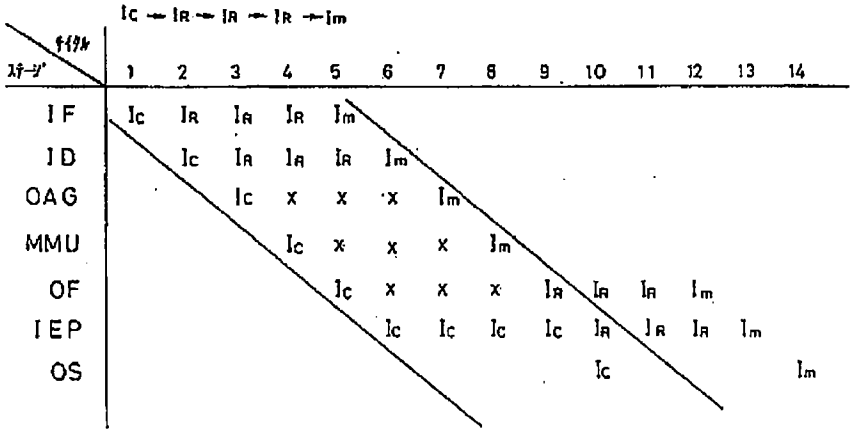
第 10 図


R/I, Rsrc (STPR)
A-mode, Areg (OAGR)
R/M1, Rsrc (CCUR)
R/M, Rdest (IEPR)
310
311
312
313
304
ID1
+1
308
322
0
15
Future F/F
323
Valid F/F
324
ID F/F
ID
314
ID2
305
317
ID3
306
315
ID4
307
OR ｹﾞｰﾄ生成回路
316
309
318
319
320
321
W
R
R
R

第 11 図


Im→IR → Im →IR →Im →IR
ｻｲｸﾙ
ｽﾃｰｼﾞ
1 2 3 4 5 6 7 8 9 10 11 12 13 14
IF Im IR Im IR Im IR
ID Im IR Im IR Im IR
OAG Im Ix Im Ix Im x
MMU Im x Im x Im x
OF Im IR Im IR Im IR
IEP Im IR Im IR Im IR
OS Im Im Im

第 12 図


Ic → IR → IR → IR → Im
ステージ
サイクル
1 2 3 4 5 6 7 8 9 10 11 12 13 14
IF Ic IR IR IR Im
ID Ic IR IR IR Im
OAG Ic x x x Im
MMU Ic x x x Im
OF Ic x x x IR IR IR Im
IEP Ic Ic Ic Ic IR IR IR Im
OS Ic Im

第13図